4

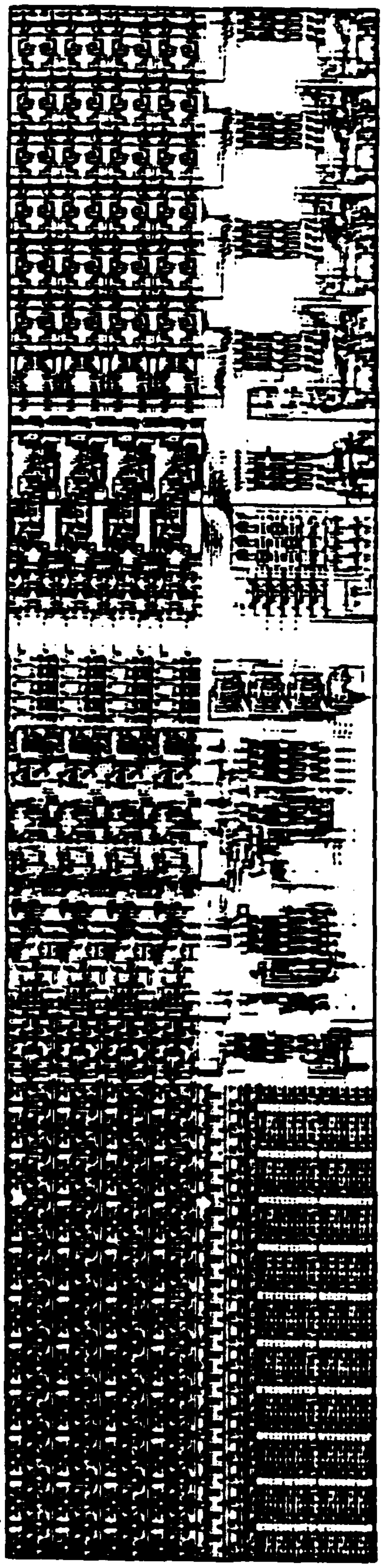
AD-A220 736

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NW-LIS-89-30-03	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Semiannual Technical Report No. 3 "VLSI Architectures and CAD"		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Northwest LIS -(Laboratory for Integrated Systems)		8. CONTRACT OR GRANT NUMBER(s) N00014-88-K-0453
9. PERFORMING ORGANIZATION NAME AND ADDRESS Northwest Laboratory for Integrated Systems University of Washington Dept. of Comp. Science, FR-35 Seattle, WA 98195		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS DARPA-ISTO 1400 Wilson Boulevard Arlington, VA 22209		12. REPORT DATE October 1989
		13. NUMBER OF PAGES 21
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of Naval Research - ONR Information Systems Program - Code 1513: CAF 800 North Quincy Street Arlington, VA 22217		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Distribution of this report is unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) NW-LIS, VLSI, synthesis, language, logic optimization, parallel simulation, switch-level simulation, normalized time, chip tester, hardware monitor, data compression, logic arrays, asynchronous circuit.		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This document reports on the research activities of the Northwest Laboratory for Integrated Systems, for the period of April 4, 1989 - November 1, 1989, under the sponsorship of the Defense Advanced Research Projects Agency, and the Office of Naval Research, under contract number N00014-88-K-0453.		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)



NORTHWEST LIS
(LABORATORY FOR INTEGRATED SYSTEMS)

Semiannual Technical Report No. 3
"VLSI Architectures & CAD"
University of Washington

November 1, 1989
LIS TR #89-30-03

Reporting Period: April 4, 1989 - November 1, 1989

Principal Investigator: Lawrence Snyder

Sponsored by:
Defense Advanced Research Projects Agency (DoD)
Issued by Office of Naval Research
Under Contract #N00014-88-K-0453

(A)

Contents

(KR)

Appendices

1 Overview of Activities

Current research in the Laboratory for Integrated Systems is focused in four areas – design specification and simulation, architecture studies, hardware testing/monitoring and integrated circuit design.

Several efforts have been pursued in the area of design specification. One such effort is *Wirelisp*, a language for specifying circuit structure that allows graphical and procedural constructs to be mixed at a fine-grained level.

Also in the area of design specification is an effort to formulate a unified representation for the behavior and structure of digital circuits. At the present time a simulator exists for the representation that checks timing constraints and performs functional simulation. Eventually this representation will provide a framework for the development of high-level synthesis and simulation tools.

In the area of design optimization is a study of logic minimization across multi-phase clocks. Our algorithms first move logic off the critical paths and across latch boundaries so as to reduce the total cycle time, then optimize for area. Our results show average delay improvements of almost 20% with area penalties of less than 12%.

Several projects have addressed switch level simulation. In one project a model for comparing synchronization strategies has been constructed. In the other project the tradeoffs between compiled and interpreted switch level simulation have been evaluated and suggest that a hybrid approach can reduce both setup and simulation times.

A recent architectural study has looked at the costs and benefits of various enhancements to existing microprocessors. The greedy algorithm has been applied to this problem and shown to be suboptimal, but nevertheless useful in reducing the number of alternatives to be considered using better (and more expensive) algorithms.

In the area of hardware testing and monitoring is a recently-constructed prototype of an inexpensive low speed tester for use with the Apple Macintosh II. The eventual goal is to create a high-quality environment for simulating and testing hardware. Also in this area is a hardware monitor for multiprocessors; this unit utilizes specialized systolic units to collect and analyze large amounts of data before storing to disk.

In the area of circuit design, a fully-testable asynchronous counter has been developed and is currently being fabricated by MOSIS. Interestingly, this counter can be tested in $O(n)$ time, in contrast to the $O(n^2)$ time required by synchronous counters.

An architecture study that shows considerable promise is a variation of the familiar Lempel-Ziv adaptive data compression scheme that permits a straightforward mapping to hardware. The essence of this scheme is a replacement strategy for the dictionary that permits continuous adapting to the input data.

2 A Unified Behavioral/Structural Representation for Simulation and Synthesis

(Tod Amon, Gaetano Borriello, Wayne Winder)

We have developed a unified internal representation for the behavior and structure of digital circuits. Behavior is defined as the union of functional and timing relationships between the inputs and outputs of a circuit, while structure is the interconnected collection of physical logic and memory elements that constitute an implementation of the behavior. By specifying precise semantics for this representation we provide a framework for the development of high-level synthesis and simulation tools. Our objective is to view synthesis as a series of transformations of the graph representation, thus enabling a fully simulatable design at all stages of the synthesis process. Currently we have a simulator for the representation that checks timing constraints as well as performing functional simulation. We are also beginning the recasting of synthesis algorithms to this new framework. This work was presented at the International Workshop on High-Level Synthesis, Oct. 1989; a copy of the paper appears in Appendix A.

3 A Circuit Specification Language that Combines Graphics and Procedures

(Zhanbing Wu, Carl Ebeling)

Wirelisp is a language used to describe the structure of a circuit. The structure can be represented as a combination of graphical and procedural elements, where Lisp expressions may be included in circuit drawings and circuit drawings may be included in Lisp expressions. A number of features such as structured signals, iterators and optional parameters make the language very expressive. This work will be presented at ICCAD, Nov. 1989.

The first version of a *Wirelisp* environment has been operational for about one year. This environment includes the graphical editor *Xdp* for drawing *Wirelisp* programs, a backend analyzer which converts these drawings into *Wirelisp*, and an interpreter written in the *T* dialect of Lisp. This system has been used to design a version of the Reprogrammable Logic Array chip described in section 11.

We have recently finished work on several enhancements. A commonly used set of graphical symbols and their corresponding *Wirelisp* definitions have been incorporated into libraries. Several tutorials and manuals have been developed. A number of enhancements have been made to the graphical frontend *Xdp* that facilitate the editing of hierarchical drawings. *Wirelisp* is currently being exercised by students in a graduate-level digital design course.

4 Timing Optimization of Multi-phase Logic

(Karen Bartlett, Gaetano Borriello, Sitaram Raju)

Timing optimization is used to reduce the cycle time of a synchronous system. In the case of complex integrated circuits, the design often uses multi-phase clocking methodologies. Most current logic synthesis and optimization systems only optimize the logic between register or latch boundaries for single-phase systems. Retiming algorithms exist to relocate registers and form an equivalent circuit with the shortest cycle time [1]; however these algorithms are not optimal for multi-phase logic and are only applicable when all phases are of the same width. Recently, optimization approaches have been proposed that move logic across register boundaries [2, 3]; however, these approaches use retiming algorithms as subroutines and therefore cannot support general multi-phase logic.

Our system is oriented towards circuits with two or more clock phases. The phases are not required to be of the same duration. Input and output signals are specified as being valid or required to be valid during any subset of the phases available.

We are using an existing logic synthesis system (MIS) to perform timing optimization on the logic and define the time for each phase. Our algorithms first move logic off the critical paths and across latch boundaries so as to reduce the total cycle time. This is then followed by an area optimization step in which logic is merged to reduce area as long as the critical path logic can still be implemented within the same cycle time.

Our algorithms yield improvements that are 10% better than what is achievable using only combinational logic optimization tools that do not move logic across latches. Furthermore we have achieved 65% of the improvements possible in the most idealized case. Results for simple two-phase circuits show average input to output delay improvements of almost 20% with area penalties of less than 12%. For a four-phase controller used in the SPUR processor our algorithms yield an improvement in cycle time of 21% with an area penalty of 21%. A complete description of this work is found in Appendix B of this report.

[1] C.E. Leiserson, F.M. Rose, and J.B. Saxe, "Optimizing Synchronous Circuitry by Retiming," *Proceedings of the Third Caltech Conference on VLSI*, March 1983.

[2] S. Malik, E. Sentovich, R. Brayton, and A. Sangiovanni-Vincentelli, "Retiming and Resynthesis: Optimizing Sequential Networks with Combinational Techniques", *Proceedings of International Workshop on Logic Synthesis*, May 1989

[3] G. De Micheli, "Synchronous Logic Synthesis", *Proceedings of International Workshop on Logic Synthesis*, May 1989

5 Comparing Synchronization Strategies for Parallel Logic-Level Simulation

(Mary Bailey, Larry Snyder)

We have looked at various asynchronous simulation techniques for parallel logic-level simulation and applied a formal model to this problem. Three different strategies were analyzed: synchronous, conservative asynchronous, and optimistic asynchronous. Both variable delay and unit-delay timing models were considered. We found that with unlimited processors the strategies can be ordered: optimistic $\leq$ conservative $\leq$ synchronous, for both variable-delay and unit-delay timing. When fixed numbers of processors are available, the above ordering may no longer hold: in particular, the conservative asynchronous strategy may be better than the optimistic asynchronous strategy. We also were able to prove that if the event evaluation times are equal, the execution times for the conservative asynchronous strategy and the synchronous strategy are equal for unit-delay timing. This work will appear in *Proceedings of ICCAD*, Nov. 1989; it is also found in Appendix C of this report.

6 A Hybrid Compiled/Interpreted Approach to Switch-Level Simulation

(Craig Anderson, Gaetano Borriello, Larry McMurchie)

We have developed a switch-level simulator that combines the speed of compiled simulation algorithms with the flexibility and fast set-up times of interpretive schemes. Compiled simulation is fast for simple subcircuits and slow for certain complex ones. Also, it must perform extensive reanalysis after even a small change in a circuit. Interpretive schemes have a fast set-up time, but are slow in simulating simple circuits because of the overhead of the interpreted data structures. Our hybrid scheme, based on *COSMOS* and *MOSSIM-II*, uses a simple heuristic to decide which algorithm is optimal for each subcircuit, drastically reducing simulation time for a variety of common circuits.

We have looked at 3 common circuit types: a dual-ported RAM, a barrel shifter and a tally circuit. Above a certain moderate size, all three types of circuits are simulated most efficiently with an interpretive scheme; below that size the compiled approach is superior.

The hybrid approach also allows modules of a large design to be analyzed independently. By using the compiled approach on subcircuits internal to each module and the interpreted approach on subcircuits that cross circuit boundaries, the setup time for the simulation of the collection of modules is minimized. We have seen a savings of a factor of 4 on several large designs.

7 Normalized Time and its Use in Architectural Design

(Sam Ho, Tom Holman, Larry Snyder)

Improvements to microprocessors are often analyzed in terms of the benefit derived from the performance increase, but rarely in terms of associated costs such as complexity and area. We recently proposed the method of normalized analysis as a way of fairly resolving both the costs and benefits of processor additions. An example of such an analysis is to ask whether programs run faster on a multiprocessor when floating-point coprocessors are installed or when the equivalent amount of hardware is used instead for additional processor elements. We have applied the greedy algorithm to normalized analysis and proven it to be suboptimal as well as conservative. The greedy algorithm is useful, however, for selecting a subset of modifications that would be found by better algorithms, thus reducing the number of alternatives that better (and more expensive) algorithms must consider. This work was presented at the 27th Annual Conference on Communication, Control, and Computing, Sept. 1989; the paper appears in Appendix D of this report.

8 Work on a Low-cost Programmable Chip Tester

(Carl Ebeling, Neil McKenzie)

We recently completed the prototype of an inexpensive low speed functional tester for use with the Apple Macintosh II. The tester consists of two printed circuit boards: an NuBus interface board that provides a parallel interface between the Mac and an external tester board which contains the tester datapath and control, and the sockets for the device under test (DUT).

The tester is controlled directly by programs running on the Mac, which allows interactive testing and debugging of chips and boards. We have an interface to RNL which allows a chip to be tested in parallel with a simulation and the results compared. We are also planning an interface to the Capilano simulation environment. The tester can be used either from MacOS or A/UX.

The current implementation of the tester uses 6 Xilinx logic cell arrays (XC3020-70 LCA) for the datapath, which provides 128 bits of parallel I/O to the the DUT. Each test pin can be programmed dynmically as either input or output, which allows the tester to be configured via software to match any chip interface. The only setup required are jumpered connections for power and ground. The tester control is implemented using a separate Xilinx chip along with several MSI counters. Chips containing dynamic state can be tested either by dedicating the Mac to the tester or by using static RAM provided by the tester to run a set of buffered test vectors.

The speed of the tester depends on the size of the chip being tested and the program drivıng the tester. Buffered test vectors run at about 500KHz, while program driven testing depends on the speed of the program. The tester has a loop capability which allows the use of a scope. There is also a programmable delay on the clock which latches the DUT outputs, allowing some rudimentary speed estimates.

The datapath is easily extendable through the use of additional Xilinx chips. The number of test vectors that can be stored may be extended via larger static RAM's. The base configuration of the tester is about $500 and we plan to make the design and PC boards publically available.

9 Hardware Assist for Performance Evaluation of Multi-Processors

(Craig Anderson, Kathy Armstrong, Gaetano Borriello)

This work addresses a serious problem in performance evaluation, namely the large volume of data that must be collected for proper studies. The kernel of the idea is to implement specialized systolic units that analyze the data as much as possible before storing to disk or memory. In this manner, most of the analysis algorithms can be pushed into hardware additions to standard logic analyzers, thus permitting the collection of data over longer runs or the use of smaller disks and memory for collection. We have designed a chip that implements a small set of possible analysis algorithms. This chip looks like a programmable cell array where much more space is devoted to state than logic - the exact opposite of what current devices provide. The challenge in this project is developing ways of recasting popular analysis programs, such as cache simulators, into a systolic pipeline that can be implemented using the functions available on the chip.

10 Hardware-Based Data Compression

(Gaetano Borriello, Suzanne Bunton, Richard Ladner, Ken Whaley)

The goal of this project is to construct a hardware data compressor with the following properties: it must consume one input every "cycle", where a cycle is bounded by 100ns; an "input" must be as wide as possible, our ultimate goal being 128 bits or more; it must effectively compress all types of data (i.e. the algorithm must be universal); the implementation must be contained within a small board.

During the last six months we have been looking at the feasibility of a hardware implementation of the Lempel-Ziv algorithm. Lempel-Ziv has been implemented both in hardware

(HP's chip, MAGIC) and software (UNIX "compress") and is well-known for its raw speed and effectiveness on all types of data.

The UNIX "compress" implementation is based upon Terry Welch's 1984 variation of the Lempel-Ziv method (LZW). One flaw lies in the fact that the dictionary ceases to adapt after it fills up. Obviously, it is important that a universal hardware data compressor adapt to the input stream continually, since the memory size is fixed and input streams can be arbitrarily long. A long input stream in which the first several words are not representative of the rest of the data will cause LZW to perform poorly. Our theoretical result is a new dynamic data structure which enables the dictionary of a Lempel-Ziv scheme to adapt continually, thus avoiding this problem entirely.

At this point the algorithm design using this adaptation scheme is correct and complete. Compressor and decompressor simulators have been completed and reflect register transfer descriptions of the circuits. Proofs of worst case performance have been outlined. The simulators have been used on a small scale to compare the performance of our algorithm to LZW (it's at least as effective, using the same size of dictionary).

The prototype we are building includes a custom CMOS chip and off-the-shelf RAM. Our projected data rate is 14MB/sec, a factor of 10 improvement over other implementations.

11 Reconfigurable Logic Arrays

(Bill Barnard, Robert Condon, Carl Ebeling)

Work is continuing on a Reconfigurable Logic Array(RLA) chip. In its simplest form, an RLA is a PLA with an added state bit for each intersection in the AND and OR planes. Earlier this year we investigated several designs for the basic cell and fabricated the most promising of these designs. Currently we are investigating different approaches to incorporating several logic blocks on a chip and dynamically configuring their interconnection. We are also looking at several target applications to determine the appropriate parameters for the size and functionality of the logic blocks.

This project has also served as a test vehicle for some of the tools we have developed over the past two years, most notably the circuit specification language *Wirelisp* and the layout generation language *WIN*.

12 Testing of Asynchronous Circuits

(Gaetano Borriello, Jerry Carson)

Designing testable asynchronous circuits is viewed as a challenge that must be surmounted before asynchronous design can become practical. We are beginning to investigate asynchronous design methodologies and how they can be modified to handle design-for-testability. We have recently completed the physical design of two versions of an asynchronous n-bit counter.

The testable counter incorporates a scan path utilizing the state storage in the counter cells, whereby the counter is converted into a 2n-bit shift register with the counter's **request** input also being used as the shift register input. The only observable outputs are the **acknowledge** and **carry-out request** signals. The counter utilizes two-cycle (transition) signaling and guarantees that new output values are available before **acknowledge** is toggled.

Two 16 bit counters, one with the scan-path and one without, are currently in fabrication (2.0 n-well CMOS) and will be used to experimentally verify the analysis. Simulations with *splice3* indicate the counter will run at an average speed of approximately 50MHz. When compared to the base cell, the testable design is achieved with a 15% increase in transistor count (from 52 to 60), an increase in chip area of approximately 6%, and a reduction in circuit speed of 7.1%.

Interestingly, it is possible to test this asynchronous counter in $O(n)$ time while a synchronous counter requires $O(n^2)$ time.